

SOFTVERSKA PLATFORMA HIDRO-INFORMACIONOG SISTEMA DRINA

Nenad GRUJOVIĆ, Nikola MILIVOJEVIĆ
Mašinski fakultet u Kragujevcu

REZIME

U radu se razmatra arhitektura hidro-informacionog sistema Drine sa odgovarajućim softverskim rešenjima. Prikazane koncepcije su osnovna specifikacija softverskih tehnologija koje su korišćene u razvoju sistema. Pored rešenja koja su implementirana, analizirana su, gde je to bilo moguće, i alternativna rešenja od kojih će neka biti primenjena u daljem razvoju. Posebna pažnja je posvećena sadašnjim i budućim rešenjima čija je uloga da sistem približe međunarodnim standardima i daju potporu harmonizaciji sa aktuelnim svetskim softverskim proizvodima.

Ključne reči: softver, baze podataka, .NET, ADO.NET, SVG, GIS.

1. UVOD

Za razvoj sadašnje verzije softverske platforme hidro-informacionog sistema Drina korišćene su najnovije tehnologije, kako u procesu projektovanja sistema, tako i u procesu pisanja aplikacija i implementacije baze podataka.

Projektovanje arhitekture sistema i izbor odgovarajućih softverskih tehnologija imao je za cilj kreiranje otvorene, skalabilne platforme koja će podjednako dobro raditi i u distribuiranom okruženju, što je danas najčešći slučaj. Pošto se radi o kompleksnom sistemu koji teži daljem proširivanju i usloznavanju, skalabilnost aplikacije je od velikog značaja, pa se od samog početka imala u vidu mogućnost da eksploataciju sistema može istovremeno vršiti veliki broj korisnika.

Najzad, imajući u vidu procese integracije koji su vrlo intenzivni u državama EU, i velike napore njihovih institucija da se istraživanja u oblasti korišćenja vodenih

resursa harmonizuju, arhitektura sistema i izabrane softverske tehnologije ograničavaju mogućnost daljeg razvoja i prestrukturiranja u skladu sa aktuelnim trendovima. Interoperabilnos modela, metodologija i rezultata istraživanja postaje prioritet u saradnji na međunarodnom nivou i zato je potrebno raspolagati otvorenom arhitekturom kao što je to slučaj sa hidro-informacionim sistemom Drina.

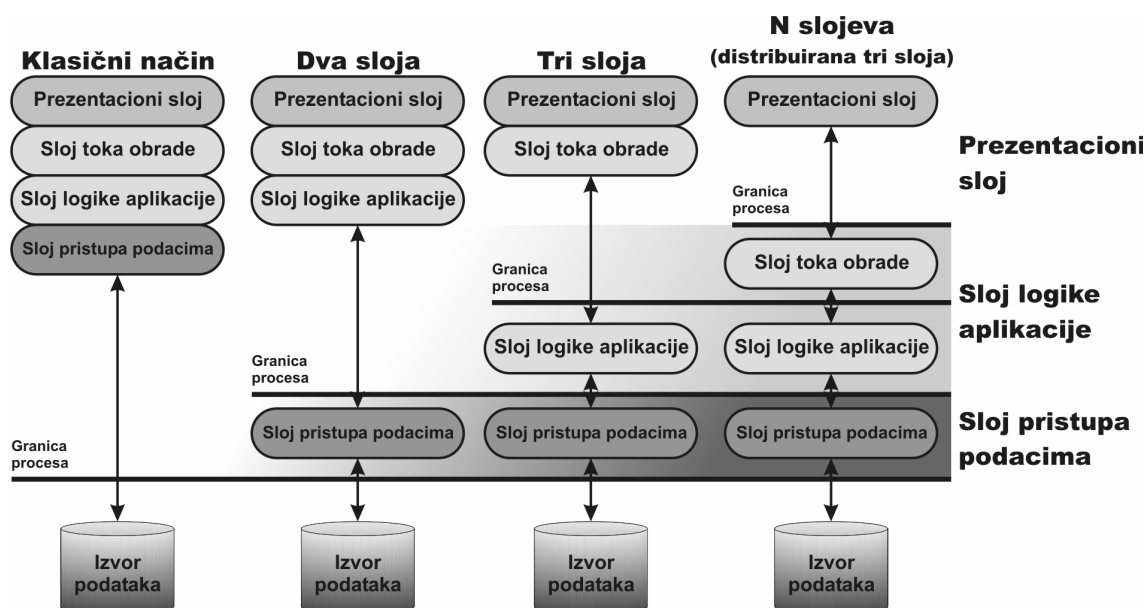
2. ARHITEKTURA SOFTVERSKE PLATFORME

Savremeni informacioni sistemi manipulišu sa ogromnim brojem podataka i principi rada koji su uspostavljeni pre nekoliko godina, danas već nisu upotrebljivi. Tako je i HIS Drina razvijen ne prema klasičnom jednoslojnom modelu, u kome aplikacija direktno pristupa podacima, već prema troslojnom modelu [1].

U jednoslojnom modelu, sav funkcionalan kôd nalazi se u samoj aplikaciji i skoro da je nemoguće napisati takvu aplikaciju koja bi se mogla održavati i razvijati bez obzira na količinu podataka i ne bi bila osetljiva na promenu, recimo, servera baze podataka.

U dvoslojnom modelu se vrši izdvajanje sloja za pristup podataka od ostatka aplikacija. Na ovaj način se obezbeđuje da veći broj korisnika može nesmetano da pristupa podacima. Više nema opasnosti od zaključavanja podataka i njihovog deljenja, jer je sda ta logika enkapsulirana u posebnom sloju.

U troslojnom modelu su jasno razdvojene funkcionalne celine: prezentacioni sloj, srednji sloj i sloj sa podacima. Prezentacioni sloj je deo aplikacije vidljiv za korisnika. Realizuje se preko Windows formi koje se prikazuju korisniku. Unutar formi su elementi korisničkog interfejsa, pomoću kojih korisnik izdaje zahteve za podacima. Srednji sloj može biti realizovan na dva načina: u vidu koda unutar formi, ili u vidu samostalnog

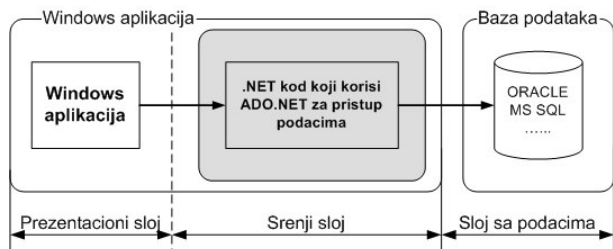


Slika 1. Razvoj arhitekture softverskih rešenja.

Web servisa, sa kojim forme komuniciraju preko SOAP poruka. U trenutnoj verziji softvera primenjen je prvi pristup.

Sloj sa podacima predstavlja ORACLE, Microsoft SQL ili bilo koju drugu bazu podataka koja je podžana od strane .NET okruženja, a veza sa srednjim slojem je obezbeđena objektima ADO.NET okruženja.

Na ovaj način se dobija na skalabilnosti aplikacije, jer se pravila koja određuju šta će se i kada raditi sa podacima, dele u dva lejera: onaj na strani klijenta i onaj na strani servera. Pošto se u ovakvoj arhitekturi postiže enkapsulacija određenih funkcija, lakši su i održavanje i dalji razvoj aplikacije [2].



Slika2. Model pristupa podacima u HIS Drina.

Na slici 2. se može videti model pristupa podacima koji je primenjen u HIS Drina.

3. SOFTVERSKJE TEHNOLOGIJE

Zahtevi koje postavlja prethodno definisana arhitektura su veliki u pogledu softverskih tehnologija. Pored toga što moraju imati visoke performanse, jer se radi o velikoj količini podataka, poseban problem je i zahtev za radom u distribuiranim sistemima. Podrazumeva se što manji pad u performansama u odnosu na aplikacije koje se izvršavaju na uobičajenim sistemima.

3.1. MICROSOFT .NET OKRUŽENJE

Microsoft .NET Framework predstavlja novu platformu za razvoj aplikacija koje rade na distribuiranim sistemima [3]. Microsoft je pokušao da postigne sledeće ciljeve:

- Da se obezbedi konzistentno OOP okruženje i kada se aplikacija prevodi i izvršava na lokalnom računaru, kada se izvršava lokalno, ali radi distribuirano ili kada se izvršava na udaljenim računarima.

- Da se obezbedi okruženje koje pojednostavljuje distribuciju aplikacija i minimizuje konflikte usled različitih verzija [4].
- Da se obezbedi okruženje koje izvršava kôd na siguran način, čak i kada se radi o komponentama koje su izradili ili nepoznati ili nepouzdati proizvođači.
- Da se obezbedi okruženje koje eliminiše razlike u performansama između prevedenog koda i skript aplikacija ili aplikacija koje se interpretiraju.
- Omogućiti programeru da iskustva stečena u pisanju određenog tipa aplikacija iskoristi i u drugim projektima, kao što je paralelan rad u Windows i u Web aplikacijama.
- Omogućiti povezivanje sa najvećim brojem postojećih tehnologija.

Postoje dve osnovne komponente .NET okruženja: Common Language Runtime (CLR) i .NET Framework biblioteka klasa. Prva komponenta CLR predstavlja «agenta» koji upravlja izvršavanjem koda pružajući usluge kao što su rad sa memorijom, manipulacija procesa u višenitnim aplikacijama, daljinsku kontrolu, a istovremeno ima preciznu kontrolu nad tipovima podataka i sve ostale osobine koje obezbeđuju siguran i robusan rad aplikacije.

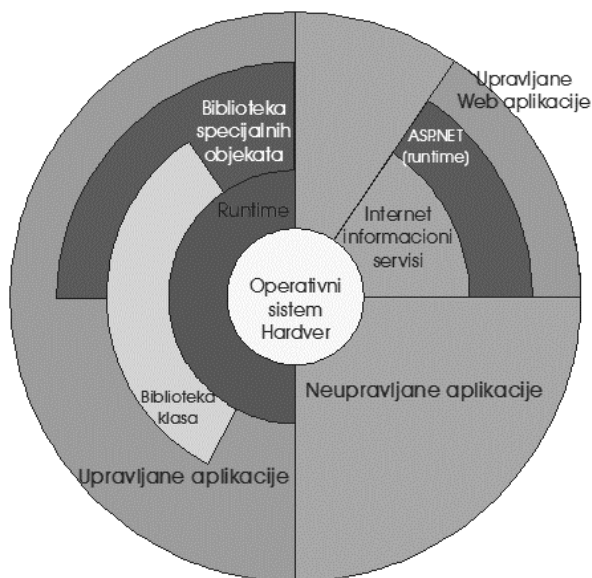
Praktično, upravljanje kodom (code management) predstavlja osnovni koncept okruženja. Kôd koji se oslanja na CLR naziva se «upravljivi kôd» (managed code), a sav ostali kôd se naziva «neupravljivi kôd» (unmanaged code). Biblioteka klasa predstavlja veliku kolekciju klasa koje se koriste pri kreiranju svih vrsta aplikacija, od tradicionalnih command line aplikacija, preko onih koje koriste grafičko okruženje Windows-a, pa sve do aplikacija koje se oslanjaju na ASP.NET i koriste Web forme i XML Web servise [5].

Komponente koje nisu upravljive mogu se koristiti sa .NET Framework-om učitavajući ga kao proces i pomoću njega pokretati upravljivi kôd [6]. Ovako se istovremeno može kreirati softversko okruženje koje koristi i upravljivi i neupravljivi kôd. Na slici 3. se mogu videti međusobne relacije svih komponenti sistema sa .NET okruženjem.

3.2. ADO.NET PRISTUP PODACIMA

ActiveX Data Objects za .NET okruženje (ADO.NET) je skup klasa preko kojih je omogućen pristup podacima .NET programeru. ADO.NET nudi obiman skup

komponenti za kreiranje distribuiranih aplikacija i predstavlja integralni deo .NET okruženja, koji omogućuje pristup relacionim, XML, i aplikacijskim podacima.



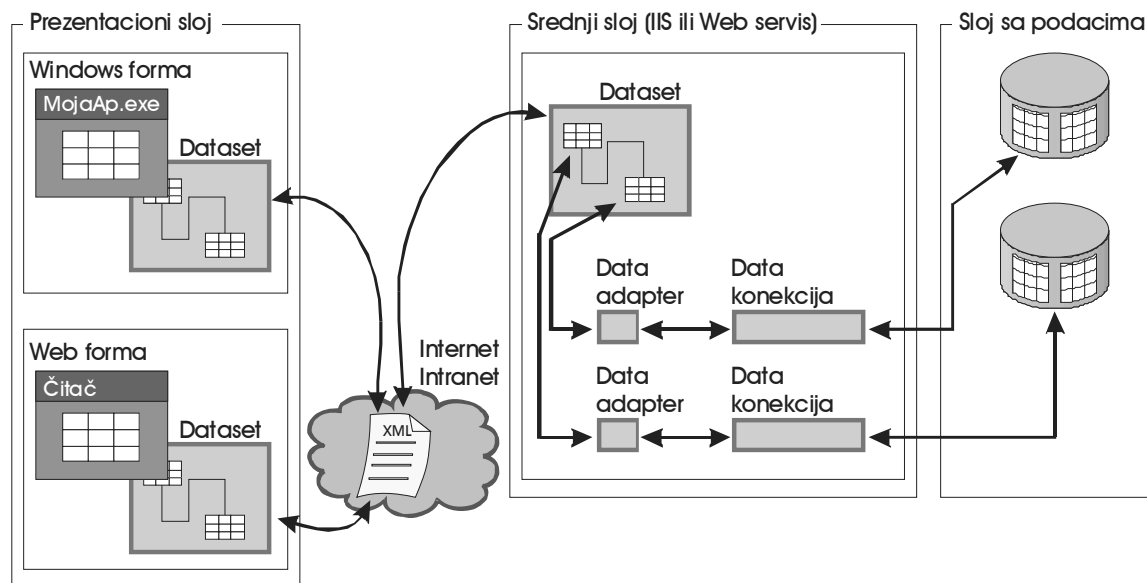
Slika 3. Povezanost sistema i .NET okruženja.

U toku razvoja aplikacija koje koriste ADO.NET, javljaju se različiti zahtevi za rad sa podacima. Ponekad će biti dovoljno prosto prikazivanje podataka na formi. Sa druge strane, može biti potrebno osmisliti način da se podaci dele sa drugim aplikacijama.

Bez obzira šta se radi sa podacima, postoje osnovni koncepti koji se moraju shvatiti vezano za pristup podacima kroz ADO.NET. Uglavnom nije potrebno znati detalje rada sa podacima (na primer, nije neophodno direktno obrađivati XML fajlove), ali može biti vrlo korisno razumevanje arhitekture podataka u ADO.NET-u, poznavanje glavnih komponenti, i načina na koji se ovi delovi uklapaju u celinu. ADO.NET ne zavisi od stalnih konekcija [7].

U tradicionalnim klijent/server aplikacijama, komponente uspostavljaju konekciju ka bazi podataka i drže je otvorenom sve dok se aplikacija izvršava. Ovakav pristup je nepraktičan, iz više razloga:

- Otvorene konekcije ka bazi podataka zauzimaju dragocene sistemske resurse. Čest je slučaj da baze mogu održavati mali broj istovremenih konekcija. Ovo opterećenje održavanja konekcija direktno utiče na umanjeње performansi aplikacije.



Slika 4. Pristup podacima kroz ADO.NET.

- Slično tome, aplikacije koje zahtevaju otvorene konekcije nisu skalabilne. Takve aplikacije mogu dobro raditi sa nekoliko korisnika istovremeno, ali ne i sa nekoliko stotina. Savremene aplikacije moraju biti skalabilne, zato što se broj korisnika može znatno uvećati za kratak period.
- U savremenim aplikacijama, komponente su po prirodi razdvojene jedna od druge. Aplikacija zahteva podatke od servera; kada server obradi podatke i pošalje ih aplikaciji, oni nadalje nemaju konekciju do sledećeg zahteva. Pod ovakvim okolnostima, održavanje otvorenih konekcija ka bazi podataka nije isplativo, jer nije poznato da li će korisnik ponovo zahtevati podatke.
- Model zasnovan na stalnim konekcijama čini teškom razmenu podataka između aplikacija koje koriste spojenu arhitekturu. Ako dve komponente treba da dele iste podatke, obe moraju biti konektovane, ili se mora osmisliti način da razmenjuju podatke.

Zbog svih navedenih problema, pristup podacima kroz ADO.NET je zasnovan na arhitekturi koja samo povremeno koristi konekcije. Aplikacije su konektovane na bazu samo za vreme potrebno da se podaci preuzmu ili pošalju. Kako baza podataka nije opterećena konekcijama koje su uglavnom neaktivne, može da opsluži mnogo više korisnika.

3.3. BAZA PODATAKA

U troslojnoj arhitekturi nije primarno pitanje servera baze podataka, već njegovih okvirnih karakteristika, i naravno, postojanja ADO.NET podrške za povezivanje na bazu. Izbor servera baze podataka, ukoliko zadovoljava određeni minimum zahteva, može uticati na performanse sistema, ali neće dovesti do prestanka rada. Pruža se čak i mogućnost istovremenog rada sa različitim serverima, što će se i dalje sa strane aplikacije videti kao «nepromenljivi» lejer sa podacima.

U slučaju hidro-informacionog sistema Drina, odabran je ORACLE server baze podataka. Glavni razlog je činjenica da ovaj server već duži niz godina opslužuje određene aplikacije u EPS-u, i da je potrebno bazu samo portovati na već postojeće servere.

Neke od osnovnih prednosti ORACLE servera su [8]:

- ORACLE RDBMS je prvi komercijalni sistem za upravljanje relacionim bazama podataka.
- Stalnim inovacijama ORACLE RDBMS zadržava vodeću poziciju u oblasti relacionih baza podataka i široko je prihvaćen kao svetski standard.
- ORACLE RDBMS je nezavisan od izbora hardverske platforme (pokriva sve postojeće hardverske platforme) i od izbora operativnog sistema.

- ORACLE omogućava efikasnu realizaciju klijent-server arhitekture za rad u distribuiranom okruženju, čime se ostvaruje maksimalno moguće iskorišćenje postojeće računarske opreme i mogućnost postepenog i parcijalnog uvođenja sa prilagođavanjem u skladu sa potrebama korisnika.
- ORACLE RDBMS je specijalno projektovan za ON-LINE procesiranje transakcija na sistemima sa velikim brojem korisnika i velikim količinama podataka.
- ORACLE RDBMS je sistem koji je otporan na greške (fault-tolerant) i omogućava ON-LINE backup i oporavak od grešaka.
- Broj korisnika sistema je ograničen samo hardverskim resursima.
- ORACLE obezbeđuje savremen pristup sa pogledom u budućnost razvoja informacione tehnologije mogućnošću paralelnog izvršavanja transakcija i radom na sistemima sa više procesora, čime se ostvaruju visoke performanse celokupnog sistema.
- ORACLE RDBMS omogućava najveću moguću otvorenost sistema: rad na 173 platforme i 28 različitih mreža.

3.4. SVG GRAFIČKI STANDARD

Jezik Scalable Vector Graphics (SVG) se koristi za opisivanje dvodimenzionalnog grafičkog prikaza u XML-u [9]. Pojam «skalabilan» se koristi za objekte koji se mogu podjednako povećavati i smanjivati. U grafičkom smislu to znači da prikaz nije ograničen na jedinice kao što su pikseli, već da je sadržaj podjednako kvalitetan bez obzira na veličinu prikaza. U Internet okruženju, skalabilnost podrazumeva da aplikacija nije osetljiva na povećanje broja korisnika koje opslužuje, ili na količinu podataka koja se obrađuje. Pošto je SVG grafički Internet standard, to znači da je skalabilan i u jednom i u drugom pogledu.

SVG dozvoljava rad sa tri tipa grafičkih objekata: vektorskim objektima, slikama i tekstem. Objekti mogu biti grupisani, mogu im se menjati stilovi, mogu se transformisati i sl. Kao napredne osobine poseduje mogućnosti alfa maskiranja, kombinovanja objekata (odsecanje, presek), primene filtera kao i korišćenje šablonskih objekata. SVG crteži mogu biti interaktivni i dinamični. Animacije se mogu definisati i pozivati unutar SVG-a predefinisanim elementima ili pomoću skriptova.

Napredne aplikacije koje koriste SVG mogu preko Document Object Model (DOM) [10] pristupiti svakom elementu u dokumentu i uticati na njegove osobine, kao i dodati potpuno nove elemente i nove osobine postojećim elementima. Na raspolaganju je i skup definisanih događaja nad dokumentom i elementima (klik mišem, pritisak tastera, kretanje miša, kraj učitavanja dokumenta itd.).

U korisničkom interfejsu hidroinformacionog sistema Drina, SVG je iskorišćen za vizuelizaciju simulacionog modela i GIS sadržaja jer poseduje mogućnost da ravnopravno manipuliše sa vektorskim prikazom i rasterskim podacima, a da pri tome omogući potpunu interaktivnost.

3.5. OWC KOMPONENTE

Office Web Components (OWC) predstavljaju minimalnu infrastrukturu potrebnu da se omogući korišćenje interaktivnih tabela, grafikona i Pivot tabela u Windows i Web aplikacijama. To su ActiveX objekti koji smanjuju potrebu za dodatnim pisanjem kôda i omogućavaju projektantima da se fokusiraju na rešenje umesto na integraciju podataka. Sadrže veliki broj tipova grafikona i sve potrebne mehanizme za prikaz i obradu podataka u tabelama. Podržan je i pristup bazama podataka direktno iz komponenti, a podaci se interno mogu pamtitu u XML obliku. Ravnopravno se mogu koristiti kako u Windows tako i u Web programiranju [11].

Postupak integracije OWC komponenti u .NET okruženju je elementaran, a pruža mogućnost prezentacije rezultata u okruženju MS Office-a koji je inače softverski paket koji poznaje praktično svaki korisnik hidro-informacionog sistema Drina.

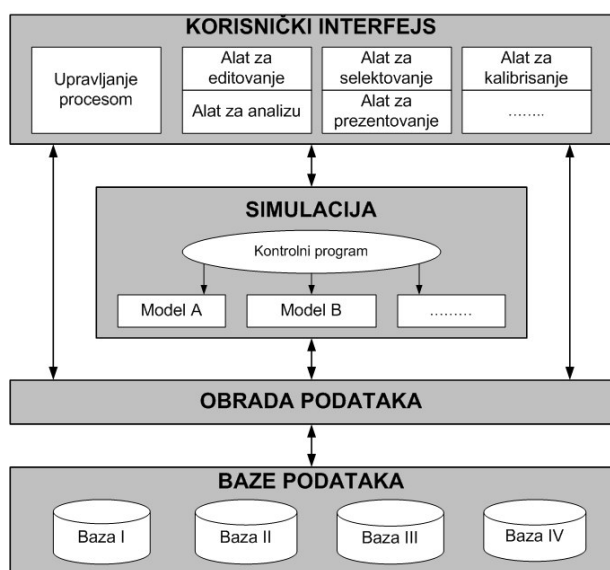
4. IMPLEMENTACIJA SVETSKIH STANDARDA

U Evropi je tokom poslednjih par decenija razvijen veliki broj simulacionih modela koji opisuju procese u eksploataciji slivova. Takođe, napisan je i veliki broj aplikacija koje se koriste pri modeliranju i simuliranju pojava u slivovima. Neke od ovih aplikacija su usko specijalizovane i verovatno će se još dugo koristiti od strane malog broja stručnjaka koji imaju posebne zahteve u pogledu softvera. Sa druge strane potreban je veliki napor da se za širi krug stručnjaka obezbede unificirani softverski paketi koji bi obezbedili

interoperabilnost u pogledu ulaznih podataka, modela, procesa simulacije i rezultata.

4.1. ARHITEKTURA SISTEMA

Zbog potreba integracije u poslednje vreme iniciran je veliki broj projekata koji za cilj imaju harmonizaciju na polju integrisanog upravljanja vodenim resursima slivova. Imajući u vidu trend prihvatanja EU standarda, i arhitektura hidro-informacionog sistema se mora približiti postojećim međunarodnim standardima.



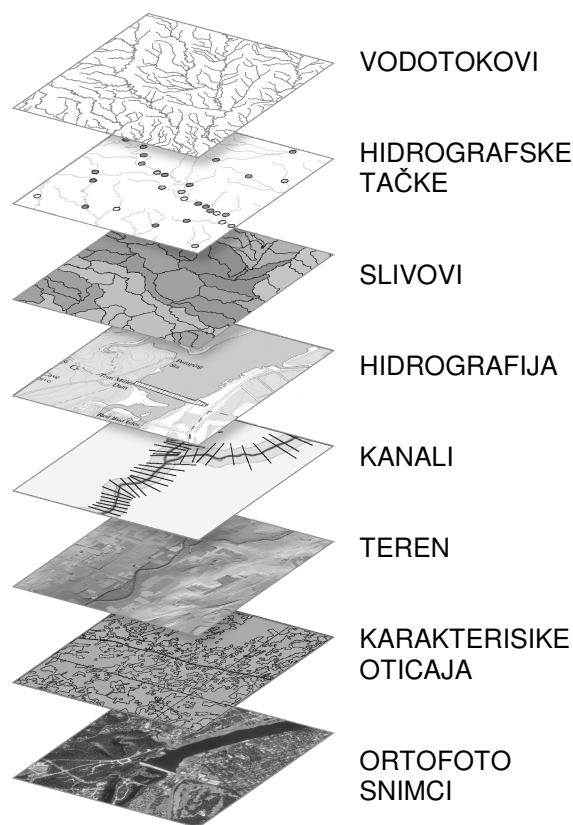
Slika 5. Prikaz Generic Framework Architecture.

Jedan od primera opšte prihvaćenih standardizovanih okruženja za razvoj hidro-informacionih aplikacija je i holandski Generic Framework Architecture (GF) [13]. Prva verzija ovog projekta je objavljena 2001. godine kao rezultat napora svih bitnih istraživačkih institucija u Holandiji da definiše arhitekturu za integraciju do tada razvijenih i korišćenih aplikacija. Do sada je izvršen reinženjering određenog broja aplikacija poput SOBEK-a i ModFlow-a i očigledan je trend prihvatanja ovog standarda [14].

Na slici 5. je dat prikaz GF-a na kome se vidi njegova troslojna arhitektura [15]. Ako ponovo pogledamo sliku 2. mogu se videti velike sličnosti sa arhitekturom HIS Drina.

4.2. GIS ELEMENTI BAZE PODATAKA

Današnji trend u razvoju softverskih platformi za podršku odlučivanju u integrisanom upravljanju slivovima jeste i sve veća upotreba GIS sadržaja. Prvobitne metode su se oslanjale na razvijanje matematičkih modela koji su zatim tarirani na dugim istorijskim serijama i nakon toga korišćeni u simulaciji i analizi. Međutim, danas se sve više modela bazira na simuliranju fizičkih procesa gde se ulazni parametri poput uticaja vegetacije, naseljenosti, načina obrađivanja zemljišta i sl., automatski izračunavaju na osnovu GIS sadržaja.



Slika 6. GIS sadržaj baze podataka.

Danas postoji nekoliko opšteprihvaćenih standarda organizovanja GIS podataka od značaja za istraživanja u korišćenju vodenih resursa, poput ArcHydro modela [16], ili opštih modela kao što je OpenGIS [17]. Organizacija podataka u bazi HIS Drina najbliža je ArcHydro modelu (slika 6).

4.3. INTEROPERABILNOST ULAZNIH PODATAKA I REZULTATA

Svi podaci se u arhiviraju u određenom formatu, i danas ih postoji na hiljade iako ih se samo oko stotinu koristi svakodnevno. Prevođenje podataka iz jednog formata u drugi je zahtevan zadatak, pa u nekim slučajevima i skoro nemoguć. Stoga je bitno predvideti mogućnost podrške najznačajnijih formata zapisa kako ulaznih podataka tako i rezultata.

Što se tiče ulaznih podataka oni su najčešće različito formatirane tekstualne datoteke koje sadrže merenja padavina, temperatura i sl. u obliku vremenskih serija. Za interoperabilnost softverskog paketa bitno je podržati najrasprostranjenije formate.

Kao najopštije rešenje nameće se upotreba XML-a (eXtensible Markup Language) kao univerzalnog standarda za opisivanje strukturiranih dokumenata. XML se može upotrebiti na dva načina: za razmenu podataka između aplikacije i baze podataka ili između više aplikacija u distribuiranim sistemima, ili kao format za zapisivanje u fajlove. Postoji nekoliko standarda koji su opšteprihvaćeni u istraživačkim krugovima kao što su:

- Geographic Mark-Up Language (GML) razvijen od strane Open Geographic Information System konzorcijuma (OGC). GML je osnovna tehnologija u razvoju geografski strukturirane Internet infrastrukture.
- Extensible Scientific Interchange Language (XSIL) kao rezultat istraživanja laboratorije Center for Advanced Computing Research na univerzitetu CalTech. XSIL je fleksibilan, hijerarhijski, proširiv jezik za razmenu objekata koji su predmet naučnog istraživanja. Posедуje skup osnovnih elemenata za opisivanje objekata i njihovih parametara kao što su nizovi, tabele i sl.
- EXCEL XML kao zapis kompatibilan sa Excel objektima. Excel zapis je naročito zanimljiv jer se u istraživanjima veoma intenzivno koristi tabelarni proračun.
- ArcHydro XML zapis koji preslikava strukturu ArcHydro baze podataka koja je ranije već pomenuta.

Hidro-informacioni sistem Drina se na više mesta oslanja na XML zapis i u procesu je definisanje XML standarda za zapis specifičnih elemenata dokumenta [18].

5. ZAKLJUČAK

Softverska platforma hidro-informacionog sistema Drina zasniva se kako na savremenim informatičkim standardima, tako i na aktuelnim standardima vezanim za istraživanjima u gazdovanju vodenih resursa. Korišćenje hidro-informacionih sistema karakterišu potreba za obradom velikog broja podataka, univerzalnosti rešenja i radu u distribuiranim okruženjima. Korišćenje .NET okruženja, XML-a, SVG-a i ADO.NET mehanizma, obezbeđuje duži radni vek i iskoristivost komponenti koje su već razvijene, kao i jednostavno uvođenje novih standarda. Dalji pravci u razvoju platforme su usmereni upravo ka prihvatanju većeg broj aktuelnih standarda i optimalnijem radu u distribuiranom okruženju.

LITERATURA

- [1] Kanalakis, J.: Developing .NET Enterprise Applications, Apress Publishing, Berkeley, 2003.
- [2] Redler, R., Rossberg, J.: Designing Scalable .NET Applications, Apress Publishing, Berkeley, 2003.
- [3] Prosis, J.: Programming Microsoft® .NET Microsoft Publishing, Redmond, 2003.
- [4] Ballinger, K.: .NET Web Services: Architecture and Implementation, Addison-Wesley, Boston, 2003.
- [5] Foggon, D., Maharry, D., Ullman, C., Watson, K.: Programming Microsoft® .NET XML Web Services, Microsoft Publishing, Redmond, 2003.
- [6] Miller, J., Ragsdale, S.: Common Language Infrastructure Annotated Standard, Addison-Wesley, Boston, 2003.
- [7] Chand, M., Talbot, D.: Applied ADO.NET: Building Data-Driven Solutions, Apress Publishing, Berkeley, 2003.
- [8] Alapati, S.: Expert Oracle9i Database Administration, Apress Publishing, Berkeley, 2003.
- [9] Cagle, K.: SVG Programming, Apress Publishing, Berkeley, 2002.
- [10] Campesato, O.: Fundamentals of Svg Programming: Concepts to Source Code, Charles River Media, 2003.

- [11] Stearns, D.: Programming Microsoft Office 2000 Web Components, Microsoft Publishing, Redmond, 1999.
- [12] Stearns, D.: Programming Microsoft Office 2000 Web Components, Microsoft Publishing, Redmond, 1999.
- [13] Blind, M.W., Adrichem, B., Groenendijk, P.: Generic Framework Water: An open modeling system for efficient model linking in integrated water management - current status, EuroSim 2001, Delft, 2001.
- [14] Blind, M.W., Adrichem, B., Groenendijk, P.: Generic Framework for hydro-environmental modelling, HydroInformatics 2000, Cedar Rapids, 2000.
- [15] Blind, M.W., Wentholt, L., Adrichem, B., Groenendijk, P., "The Generic Framework - An Open Framework for Model Linkage and Rapid Decision Support System Development", ModSim 2001, Canberra, 2001.
- [16] Maidment, D. R.: ArcHydro GIS for Water Resources, ESRI Press, Redlands, 2002.
- [17] www.opengis.com
- [18] Struve, J., Westen, S., Millard, K., Fortune, D.: Harmonit - State of the Art Review, London, 2002

SOFTWARE PLATFORM OF THE DRINA HYDRO INFORMATION SYSTEM

by

Nenad GRUJOVIĆ, Nikola MILIVOJEVIĆ
Faculty of Mechanical Engineering, Kragujevac

Summary

The architecture of the Hydro Information System of the Drina River and relevant software solutions are discussed in the paper. The presented concepts represent a basic specification of software technologies used in the development of the Hydro Information System. Apart from solutions already implemented, a number of alternative solutions are assessed and some are intended

for future development. Special attention is given to present and future solutions in order to achieve international standards and allow harmonization with world famous software products.

Key words: software, database, .NET, ADO.NET, SVG, GIS.

Redigovano 16.05.2004.